



Kahoot!



Linux Plus

for

AWS and DevOps

Session - 7

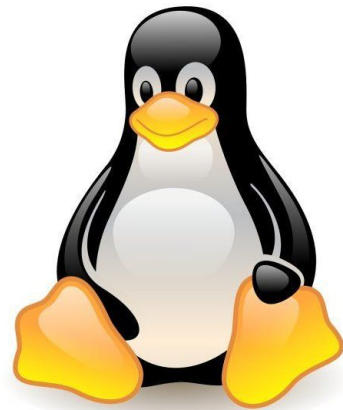


Table of Contents



- ▶ **Loops**
- ▶ **Functions**





While loops

```
while [[ <some test> ]]
do
    <commands>
done
```

```
#!/bin/bash

number=1

while [[ $number -le 10 ]]
do
    echo $number
    ((number++))
done
echo "Now, number is $number"
```

Output:

```
$/while-loops.sh
1
2
3
4
5
6
7
8
9
10
Now, number is 11
```



Until loops

```
until [[ <some test> ]]  
do  
    <commands>  
done
```

```
#!/bin/bash  
  
number=1  
  
until [[ $number -ge 10 ]]  
do  
    echo $number  
    ((number++))  
done  
echo "Now, number is $number"
```

Output:

```
$/until.sh  
1  
2  
3  
4  
5  
6  
7  
8  
9  
Now, number is 10
```



For loops

```
for item in [list]
do
    commands
done
```

```
#!/bin/sh

echo "Numbers:"

for number in 0 1 2 3 4 5 6 7 8 9
do
    echo $number
done
```

Output:

```
$/for-loop.sh
Numbers:
0
1
2
3
4
5
6
7
8
9
```



Continue and Break Statement

Infinite loop

```
#!/bin/bash

number=1

until [[ $number -lt 1 ]]
do
    echo $number
    ((number++))
done
echo "Now, number is $number"
```



Continue and Break Statement

Break Statement

```
#!/bin/bash

number=1

until [[ $number -lt 1 ]]
do
    echo $number
    ((number++))
    if [[ $number -eq 10 ]]
    then
        break
    fi
done
```

Output:

```
./infinite-loop.sh
1
2
3
4
5
6
7
8
9
```




Continue and Break Statement

Continue Statement

```
#!/bin/bash
number=1
until [[ $number -lt 1 ]]
do
    ((number++))
    tens=$(( $number % 10 ))
    if [[ $tens -eq 0 ]]
    then
        continue
    fi
    echo $number
    if [[ $number -gt 14 ]]
    then
        break
    fi
done
```

Output:

```
$/./continue.sh
2
3
4
5
6
7
8
9
11
12
13
14
15
```

Exercise 1



1. Calculate sum of the numbers between 1 to 100.
2. Print result.

Exercise 2



1. Ask user to input multiple names in a single line
2. Print “Hello” message for each name in separate lines.



Functions

```
function function_name () {  
    commands  
}
```

```
#!/bin/bash  
  
Welcome () {  
    echo "Welcome to Linux Lessons"  
}  
  
Welcome
```



Passing Arguments to Functions

```
#!/bin/bash

Welcome () {
    echo "Welcome to Linux Lessons"
    $1 $2 $3
}

Welcome Joe Matt Timothy
```

Output:

```
$/functions.sh
Welcome to Linux Lessons Joe Matt Timothy
```



Nested Functions

```
#!/bin/bash

function_one () {
    echo "This is from the first
function"
    function_two
}

function_two () {
    echo "This is from the second
function"
}

function_one
```

Output:

```
$/nested.function.sh
This is from the first function
This is from the second function
```



Variables Scope

Local variable

local variable_name=value

```
#!/bin/bash

var1='global 1'
var2='global 2'

var_scope () {
    local var1='function 1'
    var2='function 2'
    echo -e "Inside function:\nvar1: $var1\nvar2: $var2"
}

echo -e "Before calling function:\nvar1: $var1\nvar2: $var2"

var_scope

echo -e "After calling function:\nvar1: $var1\nvar2: $var2"
```

Output:

Before calling function:
var1: global 1
var2: global 2
Inside function:
var1: function 1
var2: function 2
After calling function:
var1: global 1
var2: function 2



Functions

Local variable

local variable_name=value

```
#!/bin/bash
```

```
num1=5
```

```
function add_one(){
```

```
    local num2=1
```

```
    echo "Total $(( $num1 + $num2 ))"
```

```
}
```

```
add_one
```

```
echo "Number1: $num1"
```

```
echo "Number2: $num2"
```

```
[ec2-user@ip-172-31-91-206 ~]$ ./cmd.sh
```

```
Total 6
```

```
Number1: 5
```

```
Number2:
```

```
[ec2-user@ip-172-31-91-206 ~]$
```


Exercise 3



1. Create a function named **print_age** that accepts one argument

Ask user to input his/her year of birth and store it to **local birth_year** variable

Calculate **age** using current year value from the first argument

Print **age** with a message

2. Call **print_age** function with **2023**



THANKS!

Any questions?